**МУНИЦИПАЛЬНОЕ БЮДЖЕТНОЕ ОБЩЕОБРАЗОВАТЕЛЬНОЕ УЧЕРЕЖДЕНИЕ СРЕДНЯЯ ШКОЛА СЕЛА ИЛЬИНО**

**ЛИПЕЦКОГО РАЙОНА ЛИПЕЦКОЙ ОБЛОСТИ**

**ИТОГОВЫЙ ИНДИВИДУАЛЬНЫЙ ПРОЕКТ**

**Практическая работа:**

“Создание WEB - сайта”

Выполнял работу:

Григорян Давид Жирайрович

ученик 10 класса

Руководитель проекта:

Николаев Николай Николаевич

учитель физики

Липецк

2019-2020 учебный год.

**Содержание**

**1.Введение……………………………………………………………..**

**2.Основная часть..................................................................................**

2.1. Основные составляющие сайта……………………………………………………………

2.2 Клиент – Сервер…………………………………………………………………………….

2.3. HTML……………………………………………………………………………………….

2.4. CSS……………………………………………………………….........................................

2.5. JavaScript……………………………………………………………………………………

2.6. Препроцессоры…………………………………………………………………………….

2.7. Сборщики проекта…………………………………………………………………………

2.8. Файловая структура………………………………………………………………………..

2.9. NodeJS………………………………………………………………………………………

**3.Список используемой литературы………………………………**

**4.Приложение………………………………………………………… 1.Введение**

**Объект проекта:** web-сайт.

**Предмет проекта:** разработка сайта.

**Цель проекта:**

* Написать свой web-сайт.

**Задачи проекта:**

* Сборка всех элементов web-сайта.

**Актуальность:**

* У любой современной компании существует сайт. Это один из элементов престижа, ведь именно в Интернете потенциальные клиенты будут в первую очередь искать информацию о фирме. Актуальность создания сайта состоит также в том, что если вы хотите донести информацию максимально быстро до огромного количества людей, то лучше, чем с помощью собственного сайта сделать это не получится никак. Веб-ресурс позволяет представить информацию о компании и ее товарах или услугах сжато и одновременно полноценно. Необходимо понимать, что ни один другой ресурс не даст столько преимуществ, сколько собственный сайт, будь это визитка, Интернет-магазин или любой другой веб-ресурс.

**Методы решения проблемы:**

* Найти исходные данные для web-сайта (изображения, шрифты и т д).
* Написать html, css, JavaScript файлы.

**Методы проекта:**

* теоретические – описание того из чего состоит сайт и как его сделать.
* практические – создание web-сайта.

**2.Основная часть**

**2.1. Основные составляющие сайта**

Веб-страницы могут и будут отличаться друг от друга, но все они, преимущественно, состоят из аналогичных стандартных компонентов, если только страница не отображает полноэкранное видео или игру, не является частью какого-либо художественного проекта или просто плохо структурирована:

**Заголовок** обычно это большая полоса вверху страницы, с крупным заголовком и / или логотипом. Здесь указывается общая информация о веб-сайте, не меняющаяся от страницы к странице.

**Навигационное меню** Ссылки на основные разделы сайта; обычно в виде кнопок, ссылок или вкладок. Также, как и заголовок, навигация остается неизменной на всех страницах сайта — наличие непоследовательной навигации на Вашем сайте запутает и разочарует пользователей. Многие веб-дизайнеры считают панель навигации частью заголовка, а не отдельным компонентом, но это не является обязательным требованием; на самом деле, некоторые также утверждают, что их разделение на отдельные компоненты улучшает доступность, поскольку раздельная структура будет понятнее для людей, пользующихся считывателями экрана.

**Основное содержимое** Большая область в центре страницы, содержащая, в основном, уникальный контент данной веб-страницы, например, видео, которое вы хотите посмотреть, или рассказ, который вы читаете, или карту, которую вы хотите просмотреть, или заголовки новостей и т. д. Это одна из частей сайта, которая определенно будет меняться от страницы к странице!

**Боковая панель** как правило, содержит некоторую второстепенную информацию, ссылки, цитаты, рекламу и т.д. Обычно она относится к содержимому в основном контенте (например, на странице со статьей, боковая панель может содержать биографию автора или ссылки на связанные статьи), но в некоторых случаях здесь размещают и другие элементы, например, вторичную навигационную систему.

**Нижний колонтитул (футер)** Полоса в нижней части страницы, которая обычно содержит уведомления об авторских правах или контактную информацию. Это место для размещения общей информации (например, заголовка), но обычно эта информация не является критичной или вторична для самого веб-сайта. Нижний колонтитул также иногда используется для SEO целей, предоставляя ссылки для быстрого доступа к популярному контенту.

Для реализации такой семантической разметки HTML предоставляет выделенные теги, которые можно использовать для создания таких разделов, например,

* **Заголовок: <**header**>**
* **Навигационное меню:** <nav>
* **Основное содержимое: <**main**>**, с различными подразделами содержимого, представленными элементами <artical>, <section> и <div>.
* **Боковая панель: <**aside**>**, обычно располагается внутри <main>.
* **Нижний колонтитул:**<footer>.

**2.2. Клиент – Сервер**

«Клиент — сервер» (англ. client–server) — вычислительная или сетевая архитектура, в которой задания или сетевая нагрузка распределены между поставщиками услуг, называемыми серверами, и заказчиками услуг, называемыми клиентами. Фактически клиент и сервер — это программное обеспечение. Обычно эти программы расположены на разных вычислительных машинах и взаимодействуют между собой через вычислительную сеть посредством сетевых протоколов, но они могут быть расположены также и на одной машине. Программы-серверы ожидают от клиентских программ запросы и предоставляют им свои ресурсы в виде данных (например, загрузка файлов посредством HTTP, FTP, BitTorrent, потоковое мультимедиа или работа с базами данных) или в виде сервисных функций (например, работа с электронной почтой, общение посредством систем мгновенного обмена сообщениями или просмотр web-страниц во всемирной паутине). Поскольку одна программа-сервер может выполнять запросы от множества программ-клиентов, её размещают на специально выделенной вычислительной машине, настроенной особым образом, как правило, совместно с другими программами-серверами, поэтому производительность этой машины должна быть высокой. Из-за особой роли такой машины в сети, специфики её оборудования и программного обеспечения, её также называют сервером, а машины, выполняющие клиентские программы, соответственно, клиентами.

Другими словами, клиентская разработка – это внешний вид сайта (Проводя аналогию с машиной – это внешний вид машины), а серверная разработка – это как двигатель машины.

В состав разработчиков сайта входит много частей:

1. Frontend – клиентская часть сайта (Внешний вид)
2. Backend – серверная часть сайта (Внутреннее функционирование сайта)
3. SEO – продвижение сайта (Улучшение индексирования сайта поисковыми роботами)
4. Тестировщик – проверяет код на ошибки
5. Интернет-маркетолог – продвигает сайт, но не в техническом плане, как SEO
6. Дизайнер – это разработчик внешнего вида сайта, он создает макет сайта, по которому Frontend разработчик уже реализует его.

Каждая из этих профессий узко специализируется, но есть и другие разработчики, которые обхватывают несколько частей, к примеру, Frontend developer может быть и дизайнером, и seo-разработчиком, и тестировщиком. Или есть fullstack разработчики – они занимаются одновременно и frontend и backend.

В часть Frontend входят следующие инструменты:

1. Html
2. Css
3. JavaScript
4. И много других инструментов

В часть Backend входят следующие инструменты:

1. Php
2. Python
3. И так далее

**2.3. HTML**

**HTML** — стандартизированный язык разметки документов во Всемирной паутине. Большинство веб-страниц содержат описание разметки на языке HTML (или XHTML). Язык HTML интерпретируется браузерами; полученный в результате интерпретации форматированный текст отображается на экране монитора компьютера или мобильного устройства.

Язык HTML до 5-й версии определялся как приложение SGML (стандартного обобщённого языка разметки по стандарту ISO 8879). Спецификации HTML5 формулируются в терминах DOM (объектной модели документа).

Язык XHTML является более строгим вариантом HTML, он следует синтаксису XML и является приложением языка XML в области разметки гипертекста.

Во всемирной паутине HTML-страницы, как правило, передаются браузерам от сервера по протоколам HTTP или HTTPS, в виде простого текста или с использованием шифрования.

Текстовые документы, содержащие разметку на языке HTML (такие документы традиционно имеют расширение .htm), обрабатываются специальными приложениями, которые отображают документ в его форматированном виде. Такие приложения, называемые «браузерами» или «интернет-обозревателями», обычно предоставляют пользователю удобный интерфейс для запроса веб-страниц, их просмотра (и вывода на иные внешние устройства) и, при необходимости, отправки введённых пользователем данных на сервер.

HTML — теговый язык разметки документов. Любой документ на языке HTML представляет собой набор элементов, причём начало и конец каждого элемента обозначается специальными пометками — *тегами*.

Язык гипертекстовой разметки HTML был разработан британским учёным Тимом Бернерсом-Ли приблизительно в 1986—1991 годах в стенах ЦЕРНа в Женеве в Швейцарии. HTML создавался как язык для обмена научной и технической документацией, пригодный для использования людьми, не являющимися специалистами в области вёрстки. HTML успешно справлялся с проблемой сложности SGML путём определения небольшого набора структурных и семантических элементов — дескрипторов. Дескрипторы также часто называют «тегами». С помощью HTML можно легко создать относительно простой, но красиво оформленный документ. Помимо упрощения структуры документа, в HTML внесена поддержка гипертекста. Мультимедийные возможности были добавлены позже.

Первым общедоступным описанием HTML был документ «Теги HTML», впервые упомянутый в Интернете Тимом Бернерсом-Ли в конце 1991 года. В нём описываются 18 элементов, составляющих первоначальный, относительно простой дизайн HTML. За исключением тега гиперссылки, на них сильно повлиял SGMLguid, внутренний формат документации, основанный на стандартном обобщенном языке разметки (SGML), в CERN. Одиннадцать из этих элементов всё ещё существуют в HTML 4[6].

Изначально язык HTML был задуман и создан как средство структурирования и форматирования документов без их привязки к средствам воспроизведения (отображения). В идеале, текст с разметкой HTML должен был без стилистических и структурных искажений воспроизводиться на оборудовании с различной технической оснащённостью (цветной экран современного компьютера, монохромный экран органайзера, ограниченный по размерам экран мобильного телефона или устройства и программы голосового воспроизведения текстов). Однако современное применение HTML очень далеко от его изначальной задачи. Например, тег <table> предназначен для создания в документах таблиц, но иногда используется и для оформления размещения элементов на странице. С течением времени основная идея платформонезависимости языка HTML была принесена в жертву современным потребностям в мультимедийном и графическом оформлении.

**2.4. CSS**

**CSS** — формальный язык описания внешнего вида документа, написанного с использованием языка разметки.

Преимущественно используется как средство описания, оформления внешнего вида веб-страниц, написанных с помощью языков разметки HTML и XHTML, но может также применяться к любым XML-документам, например, к SVG или XUL.

CSS используется создателями веб-страниц для задания цветов, шрифтов, стилей, расположения отдельных блоков и других аспектов представления внешнего вида этих веб-страниц. Основной целью разработки CSS являлось отделение описания логической структуры веб-страницы (которое производится с помощью HTML или других языков разметки) от описания внешнего вида этой веб-страницы (которое теперь производится с помощью формального языка CSS). Такое разделение может увеличить доступность документа, предоставить большую гибкость и возможность управления его представлением, а также уменьшить сложность и повторяемость в структурном содержимом.

Кроме того, CSS позволяет представлять один и тот же документ в различных стилях или методах вывода, таких как экранное представление, печатное представление, чтение голосом (специальным голосовым браузером или программой чтения с экрана), или при выводе устройствами, использующими шрифт Брайля.

**2.5. JavaScript**

**JavaScript** - это лёгкий, интерпретируемый, объектно-ориентированный язык с функциями первого класса, самый известный скриптовый язык для веб-страниц, но также используется во многих не браузерных окружениях. Прототипно-ориентированный, мультипарадигменный язык сценариев, который поддерживает динамический, объектно-ориентированный, императивный и функциональный стили программирования.

JavaScript запускается на стороне клиента Интернета, который может использоваться для создания/программирования того, как веб-страницы будут вести себя при наступлении каких-либо событий. JavaScript легко изучить, а также это мощный скриптовый язык, широко используемый для контролирования поведения веб-страниц.

Вопреки распространенному мнению, **JavaScript *не* является "интерпретируемым Java"**. В двух словах, JavaScript — это динамический скриптовый язык, поддерживающий прототипное создание объектов. Базовый синтаксис намеренно похож на Java и C++, чтобы уменьшить число новых концепций, необходимых для изучения языка. Такие языковые конструкции, как if, for, while, switch, try ... catch похожи на конструкции этих языков.

JavaScript может функционировать и как процедурный, и как объектно-ориентированный язык. Объекты можно создавать программно во время выполнения, путем присоединения методов и свойств или пустых объектов **во время выполнения**, в отличие от синтаксических определений классов в компилируемых языках, таких как С++ или Java. После того, как объект был создан, он может быть использован в качестве плана (или прототипа) для создания похожих объектов.

Динамические возможности JavaScript включают: создание объектов во время выполнения, переменное число параметров, динамическое создание скриптов (с помощью eval), перебор объектов (с помощью for ... in), восстановление исходного кода (программы на JavaScript могут декомпилировать тела функций обратно в исходный код).

**2.6. Препроцессоры**

**Sass** – это язык таблиц стилей, изначально разработанный Хэмптоном Кэтлином и разработанный Натали Вайзенбаум. После своих первоначальных версий Вайзенбаум и Крис Эппштейн продолжали расширять Sass с помощью SassScript, языка сценариев, используемого в файлах Sass.

Sass - это язык сценариев препроцессора, который интерпретируется или компилируется в каскадные таблицы стилей (CSS). SassScript - это сам язык сценариев. Sass состоит из двух синтаксисов . Первоначальный синтаксис, называемый «синтаксис с отступом», использует синтаксис, аналогичный Haml . [4] Он использует отступ для разделения блоков кода и новой строкисимволы для разделения правил. В новом синтаксисе «SCSS» (Sassy CSS) используется форматирование блоков, подобное CSS. Он использует фигурные скобки для обозначения блоков кода и точек с запятой для разделения правил внутри блока. Синтаксический файл с отступом и файлы SCSS традиционно имеют расширения .sass и .scss соответственно.

Pug — это шаблонизатор Html, написанный на языке JavaScript для Node.js. После интерпретации сервером синтаксис Pug превращается в Нtml код. Старое название Pug — Jade. Подробности не знаю, но у разработчика возникли проблемы с авторскими правами и проект был переименован в Pug. Так что, если встретите упоминание Jade, знайте, что речь идет о Pug.

**2.7. Сборщики проекта**

**Webpack** - это **пакет** модулей JavaScript с открытым исходным кодом . [2] [3] [4] [5] [6] Это пакет модулей в основном для JavaScript, но он может преобразовывать внешние ресурсы, такие как HTML, CSS и изображения, если включены соответствующие загрузчики. [7] webpack принимает модули с зависимостями и генерирует статические ресурсы, представляющие эти модули. [8]

webpack берет зависимости и генерирует граф зависимостей, позволяющий веб-разработчикам использовать модульный подход для целей разработки своих веб-приложений. Его можно использовать из командной строки или настроить с помощью файла конфигурации с именем *webpack.config.js* . Этот файл используется для определения правил, плагинов и т. Д. Для проекта. (Веб-пакет легко расширяется с помощью правил, которые позволяют разработчикам писать собственные задачи, которые они хотят выполнять при объединении файлов.)

Node.js требуется для использования веб-пакета.

Webpack предоставляет код по требованию, используя *кодовое разделение кода* . Технический комитет 39 для ECMAScript работает над стандартизацией функции, которая загружает дополнительный код: «предложение-динамический-импорт»

**Gulp** — это таск-менеджер для автоматического выполнения часто используемых задач (например, минификации, тестирования, объединения файлов), написанный на языке программирования JavaScript. Программное обеспечение использует командную строку для запуска задач, определённых в файле Gulpfile[2]. Создан как ответвление от проекта Grunt, чтоб взять из него лучшие практики[3]. Распространяется через менеджер пакетов NPM под MIT лицензией[4].

От Grunt отличается тем, что код задач записывается JavaScript кодом, а не в стиле конфигурационного файла[5].

На 2020-й год насчитывается более 3800 плагинов для Gulp[https://gulpjs.com/plugins/ 1].

Взаимодействия между частями программы реализуется через оператор.pipe(), выполняя по одной задаче за раз, не затрагивая исходные файлы, до конца процедуры. Это даёт возможность комбинации плагинов в любой последовательности и количестве.

Так же в Gulp усовершенствована система сборки. Это значит, что помимо запуска задач, можно также копировать файлы с места на место, компилировать и развёртывать проект в новом окружении.

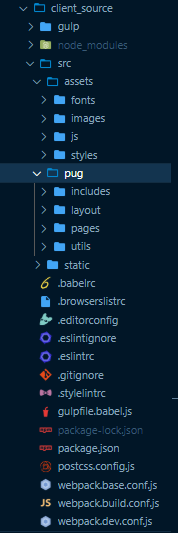
**2.8. Файловая структура**

Веб-сайт состоит из множества файлов: текстового контента, кода, стилей, медиа-контента, и так далее. Каждый проект должен иметь свою файловую структуру

Наиболее распространенные вещи, присутствующие в любом проекте сайта, которые мы создаем: индексный файл HTML и папки, содержащие изображения, файлы стилей и файлы скриптов. Давайте создадим их сейчас:

1. **index.html**: Этот файл обычно содержит контент домашней страницы, то есть текст и изображения, которые люди видят, когда они впервые попадают на ваш сайт. Используя ваш текстовый редактор, создайте новый файл с именем index.html и сохраните его прямо внутри вашей папки test-site.
2. **Папка images**: Эта папка будет содержать все изображения, которые вы используете на вашем сайте. Создайте папку с именем images внутри вашей папки test-site.
3. **Папка styles**: Эта папка будет содержать CSS код, используемый для стилизации вашего контента (например, настройка текста и цвета фона). Создайте папку с именем styles внутри вашей папки test-site.
4. **Папка scripts**: Эта папка будет содержать весь JavaScript код, используемый для добавления интерактивных функций на вашем сайте (например, кнопки которые загружают данные при клике). Создайте папку с именем scripts внутри вашей папки test-site.

Вот пример моей файловой структуры:



**2.9. NodeJS**

**Node** или **Node.js** — программная платформа, основанная на движке V8 (транслирующем JavaScript в машинный код), превращающая JavaScript из узкоспециализированного языка в язык общего назначения. Node.js добавляет возможность JavaScript взаимодействовать с устройствами ввода-вывода через свой API (написанный на C++), подключать другие внешние библиотеки, написанные на разных языках, обеспечивая вызовы к ним из JavaScript-кода. Node.js применяется преимущественно на сервере, выполняя роль веб-сервера, но есть возможность разрабатывать на Node.js и десктопные оконные приложения (при помощи NW.js, AppJS или Electron для Linux, Windows и macOS) и даже программировать микроконтроллеры (например, tessel, low.js и espruino). В основе Node.js лежит событийно-ориентированное и асинхронное (или реактивное) программирование с неблокирующим вводом/выводом.

**Список используемой литературы**

<https://developer.mozilla.org/ru/docs/Learn/HTML/%D0%92%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%B8%D0%B5_%D0%B2_HTML/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%B4%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%B0_%D0%B8_%D0%B2%D0%B5%D0%B1-%D1%81%D0%B0%D0%B9%D1%82%D0%B0>

<https://ru.wikipedia.org/wiki/HTML>

<https://ru.wikipedia.org/wiki/CSS>

<https://developer.mozilla.org/ru/docs/Web/JavaScript/%D0%9E_JavaScript>

<https://zaurmag.ru/html5-css3/html-preprotsessor-pug-jade.html>